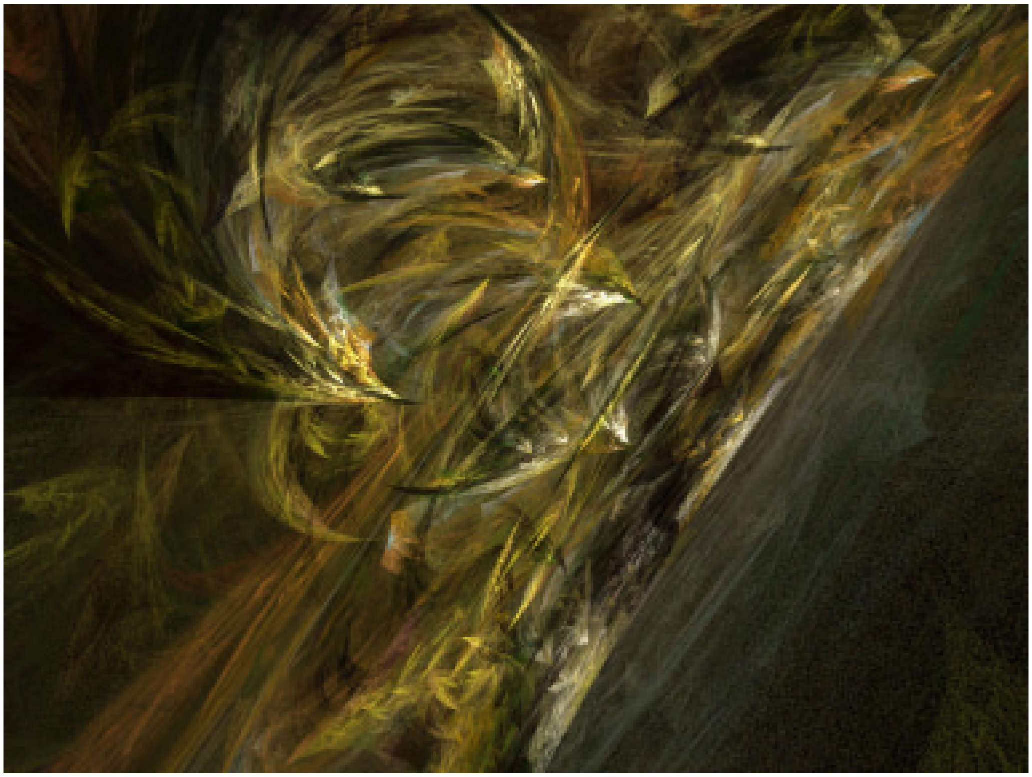
YLEM Journal

Artists Using Science & Technology

number **6** volume **23** May - June **2003**

GENERATIVE ART



A frame from one of Scott Draves work *Electric Sheep*

formerly YLEM newsletter

ASPECTS OF GENERATIVE ART
By Loren Means

This issue of the YLEM Journal presents three approaches to generative art. As I understand it, this term has come to mean art which is created by a computer program without the active intervention of the artist. Tina Yu and Paul Johnson tell how they use swarm concepts to generate choreography for dance. Bogdan Soban relates an overview of the development of his techniques for teaching his computer to create visual imagery. Scott Draves describes a web-based environment which he has created that permits other people to use his system to generate visual objects which he calls “Electric Sheep.”

Generative art is an approach which is manifesting itself in many art forms, and upcoming issues of the YLEM Journal will feature articles on generative approaches to music and literature. No attempt is made here to be definitive, since the field is growing in so many directions. We will continue to present snapshots of the bigger picture, pieces of the hologram that is called variously chance operations, artificial intelligence, artificial life, and generative art.

It is possible and viable, it must be remembered, to create generative art without the use of computers. The YLEM Journal has presented over the past few years examples of art generated by growing crystals, as in the work of Grant Elliot and Andrew Haynes, and by working with chemical processes on photographic paper, as in the work of David Berg, Ellen Carey, Susan Rankaitis, and myself. An upcoming issue of the YLEM Journal will feature an article on the cut-up techniques of William Burroughs which anticipated current work with text using computers, although the author contends that it was Burroughs’ intent to repudiate the burgeoning role of computers in society.

The term “generative” was first used extensively, it seems, to discuss the work of avant-garde novelists who were associated with the French Nouvelle Roman of fifty years ago, including Alain Robbe-Grillet, Claude Simon, Robert Pinget, Claude Ollier, Nathalie Sarraute, and Marguerite Duras. These authors were refusing to use narrative continuity as the basis for the construction of their novels, and instead were referring to outside sources such as word associations and visual imagery from paintings to help them structure their literary manifestations. The difference between the use of the term “generative” in that context and in the more contemporary usage of the term in relationship to computer-generated art seems to be in the amount of involvement the artist has in the creation of the work.

Just as some art media have a greater tendency toward abstraction than others, so some art media have taken to computer generation more quickly than others. The question of abstraction rather than the mimicking of the real world is a major unresolved conflict of Twentieth-Century art, and the role of technology in the production of art renders this conflict even more problematical, since the computer makes so powerfully available the ability to create art without the intervention of the artist. The goal of self-generated art is no longer the expression of the inner vision of the artist, but rather a manifestation of forces outside the artist which are instigated and brought into the world. The artist is then placed in the role of spectator of the work of art, a member of the audience. Yet, generative art is always a manifestation of the esthetic biases of the artist, who will suppress what is not pleasing and promote that which corresponds to the artist’s concept of what constitutes a valid work of art.

In my view, the most important technological invention for art in the Nineteen Century was the discovery of the chemical process for fixing an image on a photographic plate. This lead not only to the arts/technologies of photography and the motion picture, but also freed painting from the necessity to depict the real world, and allowed it to move into abstraction. And I perceive the most important invention for the arts of the Twentieth Century to be Benoit Mandelbrot’s discovery of fractal geometry, which manifested images of devastating beauty which could not have existed without computers. This discovery was an aspect of chaos theory, which postulates that chaos is not randomness, it is complexity, a higher order of organization.

I recently interviewed Science Fiction author Greg Bear, whose novel *Blood Music* was one of the first to explore nanotechnology, and whose latest novel, *Darwin’s Children*, explores the leading edge of biology. In an article on the New Biology on his web page, www.gregbear.com, he states: “My hypothesis: through communication by pheromones, viruses, and sexuality, and through incorporation, selection, and editing of complexes of genes by a linguistically based and computational DNA, the genomes of individuals become part of an extensive, species-scale neural network that solves problems on a much vaster scale than science has ever anticipated.”

It is this intelligence that manifests itself in the world outside ourselves that we can partner with in the creation of generative art. Rather than mimicking our perception of the outside world and our inner worlds, we can join with the forces around us and participate in this universal creativity that is bigger than our individual selves.

Tina Yu was recommended to me by Peter Bentley, author of *Creative Evolutionary Systems*. She works for a ChevronTexaco in San Ramon, CA, doing reservoir modeling, and has an extensive background in dance performance and choreography. Paul Johnson is an Associate Professor of Political Science at the University of Kansas. He works on computer models of political phenomena, and also works with agent-based modeling in such far-flung fields as dance, anthropology, and financial markets. Scott Draves also known as “Spot,” wrote an article in the May-June 2000 edition of the YLEM Journal, available at www.ylem.org. His work has been featured in *Wired* magazine, and won Best of Show in an Art, Science, and Technology show I chaired at Artisans Gallery in Mill Valley in 2001. The highlight of that show for me was when Aaron Ross arrived at the gallery to submit a piece he had created using Scott’s algorithms, and they met in person for the first time. Spot's day-job is now in the R&D department of PDI/Dreamworks. Bogdan Soban is a pioneer in generative art in Slovenia. He works in the field of e-commerce in the Research and Development department of Slovenia’s biggest casino (www.hit.si). I find this professional connection to the exploitation of chance operations fascinating, and it reminds me of Einstein’s statement in refutation of Niels Bohr’s defense of Heisenberg’s Uncertainty Principle, “The Old One does not play dice with the universe,” and of how a Latin American country erected a statue of Einstein facing a casino, so he can watch people play dice for eternity.

YLEM Forum

The last forum “Tech and Textiles” at San Francisco’s Zeum was a great success! Lia Cook, Mary Steigliz and Famous Melissa presented their ideas and work to an audience of 60 people. Special thanks to Zeum’s Exhibits Manager Lisa Dunmeyer for enabling us to hold our forum there. Be sure to check out zuem.org and their latest exhibit *Whoosh!* which opened May 24th.

Next forum:

YLEM Forum:
Computers Enhancing Community
Wednesday, July 9, 7:30 PM
McBean Theater, The Exploratorium
3601 Lyon St., San Francisco, CA 94123
The museum is not open in the evenings, so follow signs to enter.

The YLEM Forum is free, open to the public and wheelchair accessible.
Sponsored by YLEM: Artists Using Science and Technology.

Enhancing our sense of ourselves in the place where we live, enhancing the abilities of the disabled, enhancing our ability to organize our own joyful or serious community events, these are just some of the ways computers can enrich our lives.

The first speaker recruited to speak is Bonny Sherk, but more will follow! Please visit the YLEM web site for more recent news of the upcoming forum.

Bonny Sherk, founder and director of A Living Library, describes it as “themed, content-rich landscapes with integrated community programs, multidisciplinary project-based learning, and state-of-the-art communications technologies. *A Living Library* is created by all sectors of the community, particularly students, and cultivates the ’Human Garden’ by its emphasis on diversity, commonalities, participation, and inclusivity from the peoples of the world. *A Living Library* provides a practical and enchanting way to bring us all together and celebrate life.” The web site address is www.alivinglibrary.org .

How does Sherk bring this about? “*A Living Library* transforms sunken meadows and brown fields, urban sprawl and desolation, public parks and plazas, concrete and asphalt school yards, civic centers or undeveloped wastelands into vibrant and relevant multicultural community learning environments.” It makes them, she says, “highly visible public magnets“ which, in turn, are a “systemic framework and vehicle for environmental and educational transformation.”

Contact: Trudy Reagan, 650-856-9593
trudy.myrrh@stanfordalumni.org
Complete information listed at http://www.ylem.org

News

Please note, our mailing address has changed! The new mail address is:

YLEM: Artists Using Science and Technology
P.O. Box 2590
Alameda, CA 94501

We have a new YLEM member gallery show coming up opening August 7th at the Blue Room Gallery on 2331 Mission Street in San Francisco. The exhibit will be curated by Paul Mahder, Executive Director of the Blue Room Gallery. Details will be posted on the YLEM web site and at www.blueroomgallery.org.

Check out our new featured artist on out web site, Bathsheba Grossman.

Our web site ylem.org will be updated on the 28th of each month.

MY GENERATIVE ART STORY
By Bogdan Soban

Working in the area of informatics for a long time I often thought about unusual possibilities of computer technology. Developing commercial, business and production applications was persuading me that the computer could be able to do much more than reading and elaborating data and printing lists or screens. I had in mind human abilities such as intelligence, creation, thinking, etc. An eternal question was if the computer could create.

There are some areas such as processing speed, perfect memory and infallibility, where the computer surpasses a human. Naturally all these properties are not enough to make a computer creative, but they are very useful among others in simulating processes where we need to elaborate the "infinite" number of cases. To build into a program using coincidental decisions means to be closer to human behavior and creativity. Applying a good random number generator and coincidental changing seed depending on current time is a good way to come near a solution.

When the most popular personal computer, the C64, was available, I developed my first programs using the simple BASIC programming language and its random number generator. It seemed to be my invention, because I had never heard of this approach. Images appeared on the TV screen and to print them was only dreaming. It was in the early 80s and my approach was to create simple images with a computer, which amazed everybody to whom I presented my programs. In the process of creation the computer was completely autonomous to the highest possible degree.

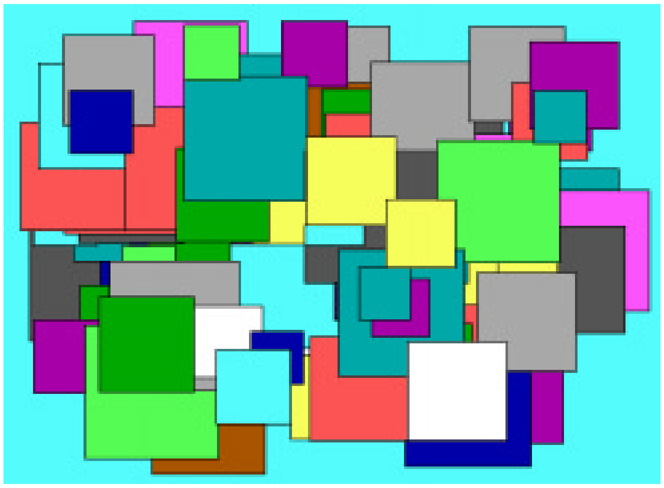


Figure 1

By means of a coincidental function the computer itself was free to choose the largest possible number of elements determining a graphic image such as type, number, shape, size, color, position on the screen etc. It always created new, unpredictable and never-repeatable images. Unpredictability was achieved by positioning in the program a series of coincidental numbers subordinated to the momentary time value when the program started. The gene scheme of the picture that determined the entire graphic image was generated in this moment. Figure 1 is a typical picture from that time.

It was a silly idea to think that a computer can create, that a computer can do something unpredictable considering the von Neumann rules of the execution of the program, step by step doing strictly coded actions. (von Neumann stated that unpredictable results could only be caused by program or data errors.) But the results that appeared on the screen had all the properties of new creation. I was persuaded that by introducing this method it could be possible to create artworks by applying other programming methods to reach the highest aesthetic level. In those times my experiences were not so rich as to realize the idea of doing art on a machine level without any kind of outside influence on the process. Discussing the basic idea with artists or programmers was a kind of adventure. Artists swore on the human soul and emotions, programmers on the determinism of the program code. But under certain cir cumstances it could be possible, was my answer, and I continued to develop my idea.

Until the year 1990 I developed a lot of simple programs but having no possibilities to print the created images on paper, I was a little bit handicapped. I engaged my friend with a camera and we made some photos from the TV screen with no great success. We tried to use the colored plotter too but not having the right software interface the paper remained white. Because I didn't find the right way to present my project in public, my creativity began to decrease.

Fast development of informational technology and its widespread use made hardware cheaper and cheaper. In the year 1991 I got my first personal computer with a color screen and the next year I bought a color printer. Having all I needed, I started to develop new programs using the GWBASIC programming language. I used colored points, lines and planes as basic elements of the image. To arrange them and to achieve aesthetic value of the image was the task of the program and the computer.

It was in the year 1995 that I participated for the first time in the Festival of Computer Art where I presented printed artworks defined as geometric abstractions (figure 2). It was much more attractive to see the growth of the image on the screen, rather than to look at prints as exposed "artworks". So the next year I participated in the Festival, with the live presentation running my program in an infinite loop. Images appeared on the screen for hours and hours, and this was one of the first presentations of generative art in our country, Slovenia.

Near the end of the year 1999 I discovered "Generative Art", a worldwide method of creating artworks very close to my basic concept. I developed my project for years and years "alone in the dark" and then in one moment I felt like I was a part of group with the same ideas. Everything appeared "deja vu". Reading papers on the Internet and finding discussions similar to my considerations in the past was the real surprise for me. In fact I had never heard of generative art before, but I passed nearly the same way of development as many others without knowing what was going on in the world.

My experiments cover the visual art area only, but discovering other fields of application of generative methods, such as architecture, industrial design, music, poetry, etc., made a great impression on me. Theory about the authorship of generative art confirmed discussions I had in the past on occasions of exhibitions of my works. It was always rather difficult to explain the complexity of the authorship of generated artwork in comparison to the absolutely clear case of an artist working on canvas.

Discovering experiences of other programmers-artists I learned a lot of new concepts and approaches for doing art with the computer. Examining closely the algorithmic approach and properties of mathematics with its aesthetic built in, I was convinced that using this method it could be possible to realize real art. Until this period I developed exclusively pragmatically-designed programs where the type of result is defined and the computer is not absolutely free to create.

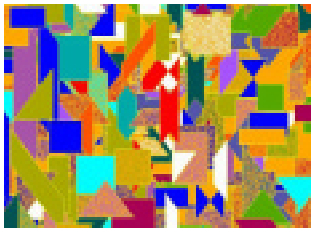


Figure 2

Taking this way I developed my first program based on algorithms using mathematic expressions and formulas. The results were surprising and better than previous results from the artistic point of view. I was sure that the basic aesthetic laws are built in mathematics because very few of the generated images needed to be thrown away. On Figure 3 is presented one

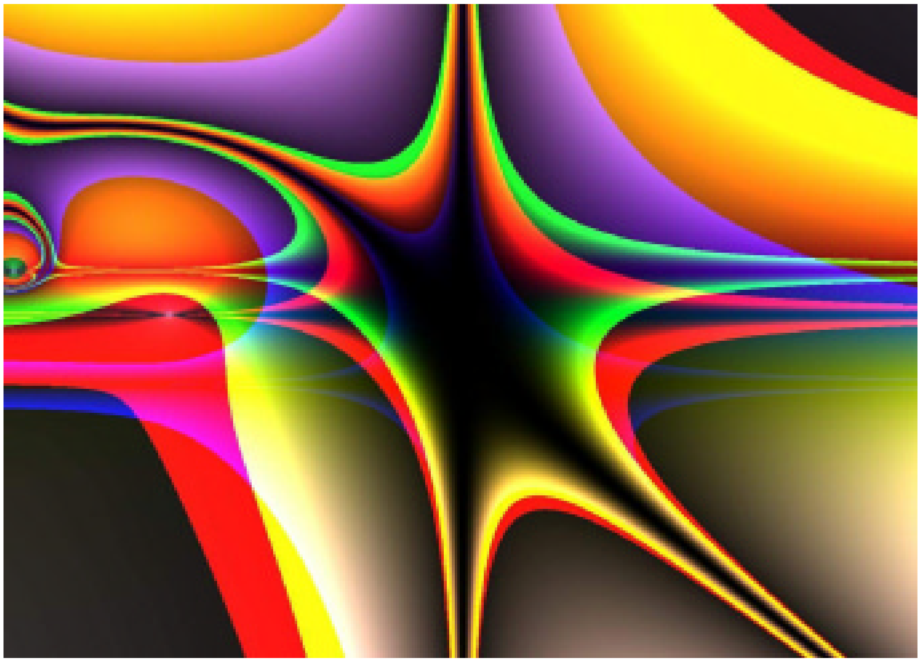


Figure 3

example of a mathematically generated picture.

Examining closely the mathematic type of programs I discovered that until the program is alive the image exists in an immense 2D space because all mathematic expressions have definite values for all values of x and y in the Cartesian coordinate system. I arranged one program that made it possible to “walk” up and down, left and right, and discover how the image looks on different distances from the starting-point of the coordinate system. No canvas or screen limits any more, such an image exists in the universe.

Using any commercial program for design, we can create images on the screen or in the memory of the computer in a limited size (screen, memory limits). Until my program is alive (when it is running or waiting for an action), it is possible to calculate the color of the pixel for each value of the coordinates, with no limits. In this way is possible "to make visible" a part of the image that is situated, for example, 1500 screens in the direction “right”, or 5000 screens in the direction “up”. The genetic code created in the start of the program is still the same and we are still on the same image, so we can "walk around the picture" and admire its appearance all over. I am still developing this algorithm because there is a small number of cases where the appearance of the image has a consistent shape very far from coordinate point zero, zero. The solution is to build correct mathematical expressions and it needs a lot of experiments. But the concept works and until now I didn't hear of similar experiments. To understand better is necessary to see how the program works (to make a live presentation).

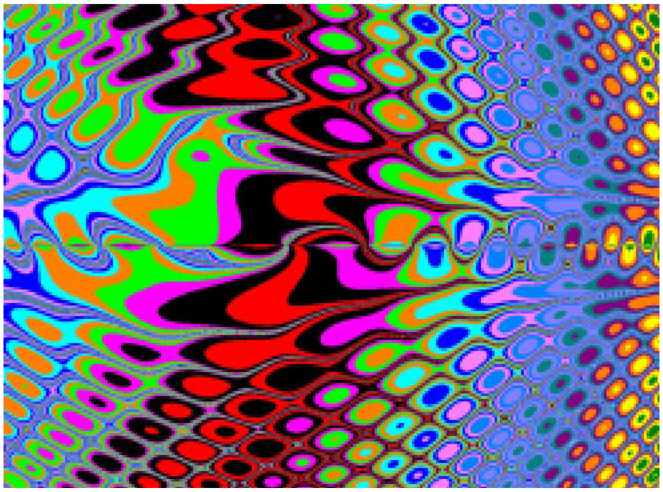


Figure 4

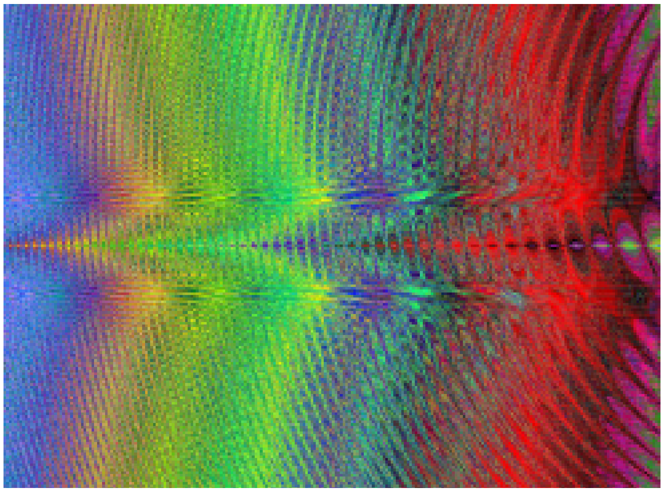


Figure 5

It is Important to apply correct mathematic expressions that can create interesting shapes far from the point (0,0). Figure 4 presents a look at the immense image in the central position and figure 5 presents a look at the same image 69 screens right from the point (0,0). This way of research is still in the beginning and I think that it could be very useful, especially in the area of practical applications of generative results.

Programming in GWBASIC caused more and more limits to the development of my project and problems on new WINDOWS versions. In the year 2001 I began to apply VISUAL BASIC, a very powerful programming language and very close to my previous knowledge. Using the new language I discovered new color possibilities

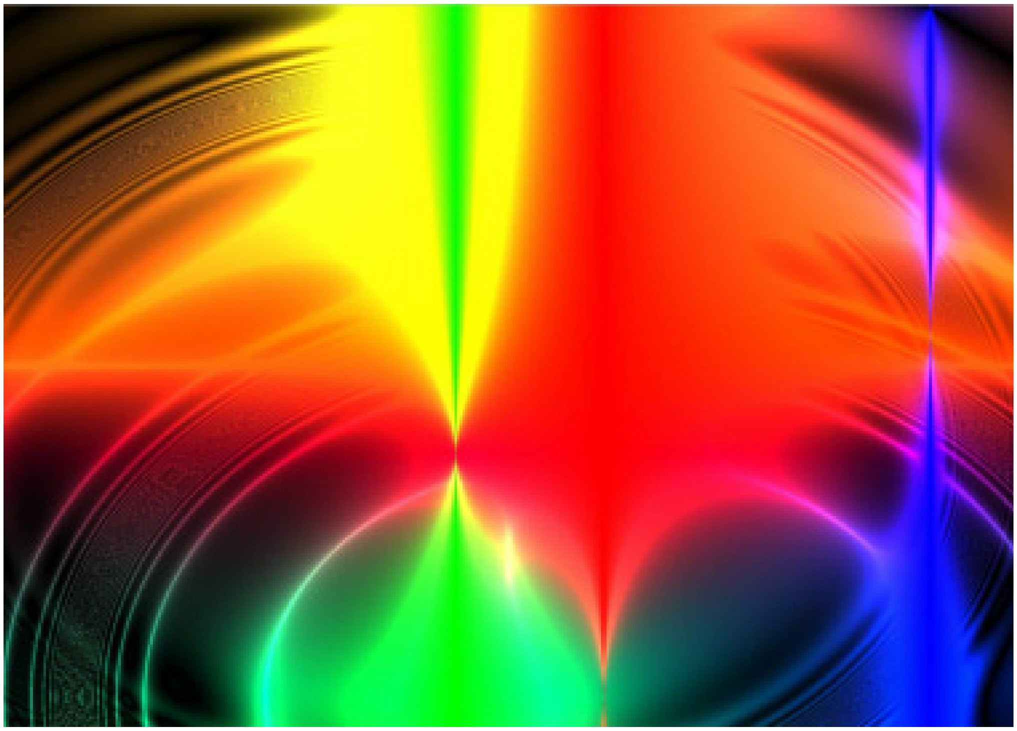


Figure 6

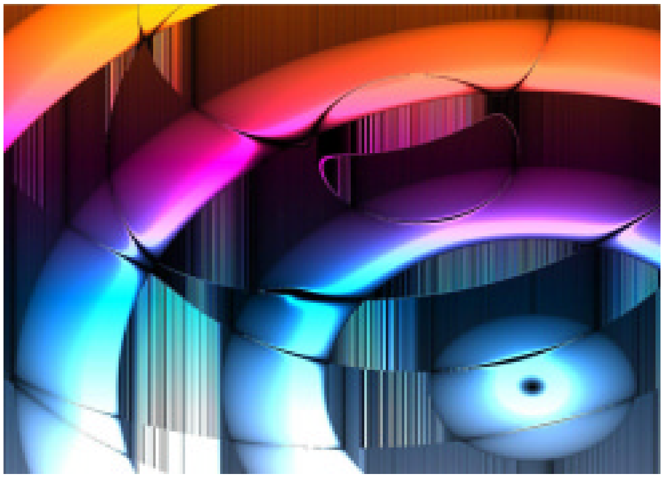


Figure 7

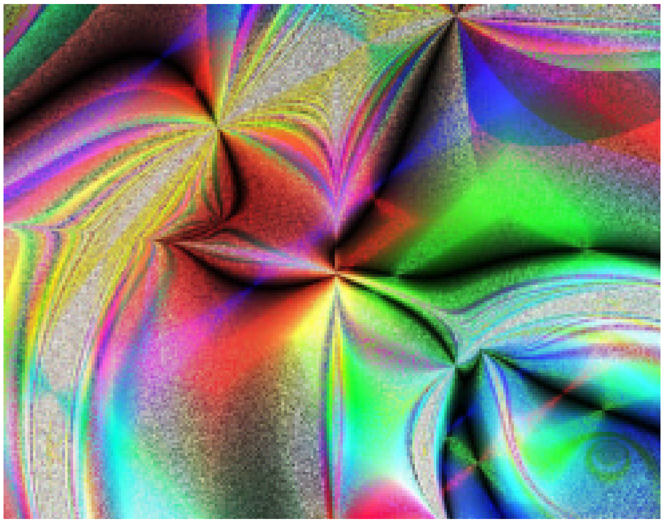


Figure 8

that had a very important influence on my recent work. I continued developing algorithms based on mathematic functions and expressions reaching a high aesthetic level of created artworks (figure 6).

Coincidental use of two different mathematical expressions on the level of the plane makes it possible to create a kind of hybrid (figure 7). The same method applied on each pixel produces an image in pointillism style (figure 8). Elaborating the created image with a cellular automata method, the computer can create colorful artworks in impressionism style (figure 9). My newest discovery is “images of parallel worlds” as I call a type of semi-realistic views on the landscape (figure 10), the results of pure mathematical algorithms.

We never know the future exactly, but I’m sure to continue with mathematical algorithms, and I’ll try to get more and more creativity from the computer without any type of outside influence. I think there are no limits on this way, the way of artificial creativity.

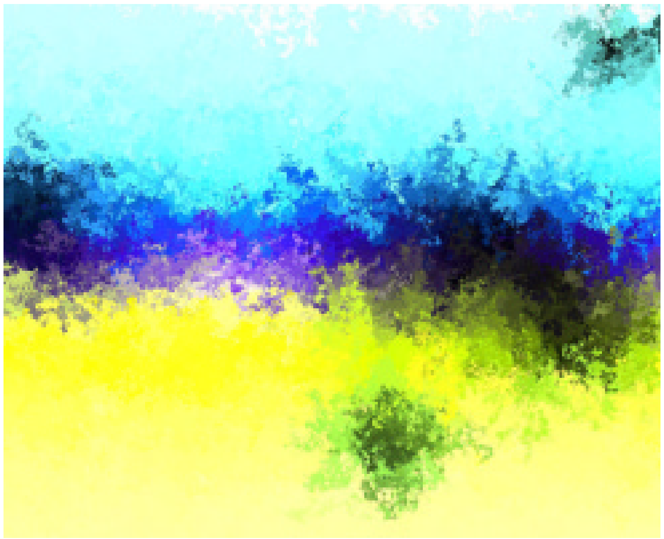


Figure 9

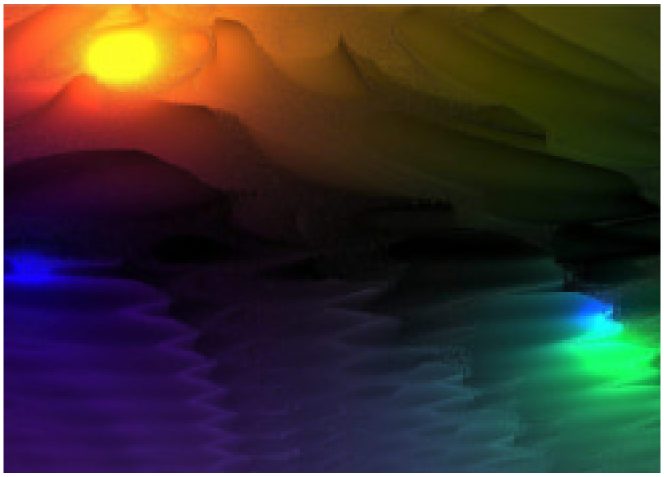


Figure 10

Tour Jeté, Pirouette: Dance Choreographing by Computers

Tina Yu[1] and Paul Johnson[2]

[1]ChevronTexaco Information Technology Company
6001 Bollinger Canyon Road
San Ramon, CA 94583
tiyu@chevrontexaco.com
http://www.improvise.ws

[2]Department of Political Science
University of Kansas
Lawrence, Kansas 66045
pauljohn@ukans.edu
http://lark.cc.ku.edu/~pauljohn

1. Introduction

Dance might be one of the most egoistic art forms ever created. This is partly due to the fact that human bodies are highly unique. Moreover, it is very difficult to record dance movements in precise details, no matter what method one uses. As a result, dances are frequently associated with the name of their choreographers, who not only create but also teach and deliver these art forms with ultimate authority. Such tight bonds between a dance and its creator gives the impression that dance is an art that can only be created by humans.

In this work, we explore the possibility that a more-or-less rigid set of computerized rules can generate interesting suggestions for dances. We have approached the challenge of demonstrating the workability of computer-generated choreography in two stages. First, we have implemented an agent-based computer model which represents the dancers as self-contained individuals who perform sequences of dance movements. These movements are generated by computers through a random selection of possible steps defined by a set of aesthetic rules. At the second stage, the dance sequences are animated with a small degree of improvisation. The generated dance choreography is very interesting and has received encouraging feedback from professional choreographers. This paper describes the work and show some of our results.

2. Swarm Mult-Agent System

A multi-agent system is a computer model in which individuals (agents) are represented by self-contained computer objects. These objects behave according to a set of rules, which dictate where they go and which steps they perform. Our implementation of the model is done with the Swarm simulation toolkit, a general purpose simulation modeling library that was first developed at the Santa Fe Institute [7] and is now under the supervision of the Swarm Development Group (http://www.swarm.org). The dance program we have implmented is freely available at http://lark.cc.ku.edu/~pauljohn/Swarm/MySwarmCode/Dancer.

In the Swarm Dance model, the dance floor is a grid with size 8 by 10. Each dancer keeps the information of the current location and the step just completed. Initially, all dancers are standing at the locations that we specify. After that, computers select the steps for each dancer randomly from a set of possibilities defined by the given aesthetic rules. If the step will lead to collision between dancers, or the step will take the dancer off the stage, then a differet step will be chosen. This trial-end-error process continues throughout the dance creation process.

In the current implementation, we only provide ten simple dance steps for comput-ers to create choreography. Most of these are ballet steps because ballet has established vocabularies that are easy to understand. In addition, we added floor steps (such as roll forward) to give the piece some modern flavor.

For this piece, we use four dancers. At the beginning, they stand at the positions we specified. After that, each dancer chooses the next step from the menu of possible steps that we have designed for aesthetic purposes (Table 1). The selection is random; hence allow novel dance patterns to be created.

Table 1. The 10 Dance Steps and Its Next Step May Be

Step No.	Step Name	Next Step May Be
0	Stand	0,1,2,3,4,5,6,7
1	Run	0,1,2,3,4,5,6,7
2	Roll Forward	2,8,9
3	Back-side-coupé	0,1,2,3,4,6
4	Soutenu Turn	0,1,2,3,4,5,7
5	Balancé	0,1,2,3,4,5,7
6	Tour Jeté	0,1,2,3,4,5,6,7
7	Par de chat	0,1,2,3,4,5,7
8	Swing leg side	8,9
9	Stand up	0,1,2,3,4,5,6,7

2.1 Scheduling

In a Swarm program, time passes in small, discrete units called *timesteps*. From a programming point of view, one of the most interesting aspects of this project is that not all dancers begin new steps at each time point. Rather, some steps take several timesteps and a new dance step can be selected and initiated only after the previous one is completed. We have dealt with this through an approach that is known in Swarm as dynamic scheduling, meaning that the dancers choose their steps according to their personal time frames. This differentiates Swarm from the more-usual agent-based model in which each agent chooses a new action at each time point [3].

2.2 Position Changes

Another programming task is on the changes of dancers' positions. Some of the dance steps are done in place, without causing the dancer to change positions. Others cause the dancer to move, possibly from left to right, or possibly in any direction. For the sake of simplicity, we divided the possible directions into 8 groups. We numbered the directions as indicated here:

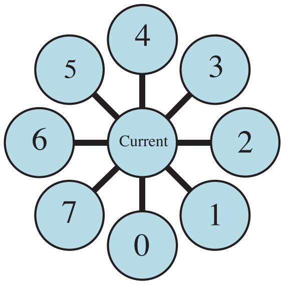


Figure 1. The eight possible movement directions.

Some of the steps can only be done in a single direction, such as roll forward, while some, such as the soutenu turn, can go either to the left or the right. Others, like run, can go in any of 8 directions. The dance steps, along with the timesteps required finishing them, the distance of the step they cause, and the number of direc-tions, are illustrated in Table 2.

Table 2. Movement Information for the Dance Steps.

Step No.	Step Name	Timesteps Required	Distance	Possible Directions
0	Stand	1	0	1
1	Run	3	1	8
2	Roll forward	12	1	1
3	Back-side-coupé	3	0	8
4	Soutenu Turn	6	1	2
5	Balancé	6	0	1
6	Tour Jeté	3	1	8
7	Par de chat	3	1	2
8	Swing leg side	1	0	2
9	Stand up	3	0	1

Once the choragraphy is completed, we animated the dance using Life Form software [4]. Life Form has been used by other choreographers, such as Merce Cunningham, to explore body shapes and movements for creating new dances. In our work, we use the software to generate dance animation, which provides a visual image of the dance movements created by Swarm.

Dance animation involves posing and positioning the dancers at various points in time. For example, at time zero a dancer may be in a crouched position preparing to leap and one second later she may be in mid-air. There are three phases in composing dance animation using the software.

At the first phase, the figure poses (shapes) of a dance step (e.g. jump) are defined. We use the Figure editor (Figure 2) to specify these poses.

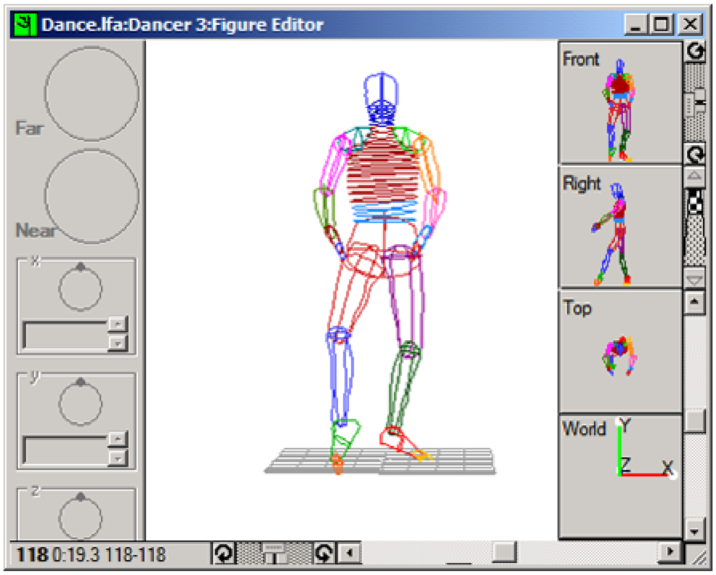


Figure 2. The Figure Editor Window.

Since a dance step is a sequence of figure poses taking place in a series of time frames, the second phase is to compose these different poses together to produce the animation of the dance step. This is done using the Timeline editor (Figure 3).

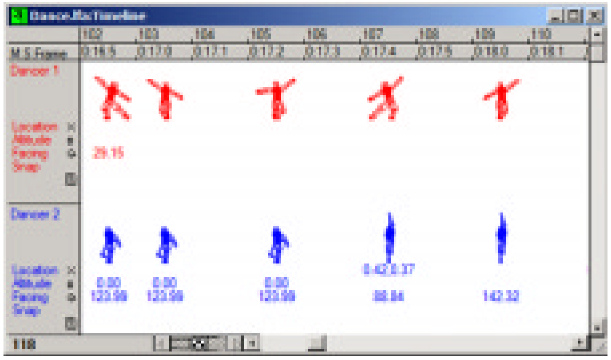


Figure 3. The Timeline Editor Window.

We generated animation for each of the ten dances steps using the Figure and Time-line editors and stored them in an animation library. Among them, the roll forward step is the most difficult to animate. Although we have allocated 12 timesteps for this movement, the detailed body shape is not easy to capture. In dance, a position “grows” to next position; there are many in-between shapes that are needed to move a dancer smoothly from one position to the next. Although the software provides interpolation methods to calculate the in-between shapes, we find them insufficient for complex movements.

Once the animation of all dance steps is completed, we are ready to put the dance steps together to compose a dance piece. The third phase is to animate the dance step sequences generated by Swarm using the Stage Window (Figure 4). We have selected J. S. Bach’s Cello Suits No.1 for this dance since it matches the abstract style of piece.

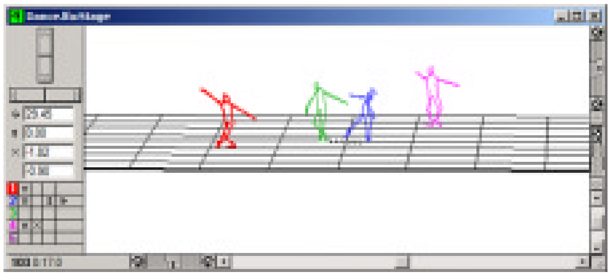


Figure 4. The Stage Window.

Animating the dance steps on stage is like giving the dance performance. We allow the dancers to have some degrees of improvisation according to the rhythm of the music. For example, the in-air position of par de chat is synchronized with the high bit of the music. Another example is if two dancers have selected the same step, they will start the step at the same time. This gives a harmonious vision on stage.

Another editing we have done is the introduction and the ending sections. While playing the animation, it is apparent that roll forward is the most selected step. We like to use this theme step to close the piece. One simple way is to have all dancers doing this step in a straight line. In order to make this happen, all dancers have to walk to the designated straight line positions. We therefore compose the closing sec-tion with a walking step followed by the roll forward step. Additionally, we added the walking step at the very beginning of the piece, with a slight variation that the walk-ing takes place on the spot, i.e. dancers do not change their position.

We have considered using computers to create the introduction and closing sections. However, computers can not generate anything exciting given only the roll forward step (which only has one direction). In order to generate interesting dance pattern, computers need more than one choice for random number to take effect. We will add variation on position and speed for roll forward step in our future work.

The completed animation (http://www.improvise.ws/Dance.mov.zip) has presented to two professional chorographers. Both Charles Anderson of Company C. and Viktor Kabaniaev of Diablo Ballet are very impressive by it and encourage us to continue the work for future stage produciton.

4 Conclusions

Dance choreographing is a creative process normally done by humans. Our prelimi-nary investigation of computer-generated choreography has shows some encouraging results:

1. The agent-based Swarm model is able to create interesting dance movements based on the provided aesthetic rules. Our implementation is flexible and easy for new rules to be added later. For examples, we are devising new rules that coordinate dancers in selecting next dance steps to induce coherent dance patterns. We are in-terested in exploring the possibility that computers can develop unique dance styles just like human choreographers do.

2. The computer animation process gives an opportunity for human intervention to produce more coherent results. Creative activities, such as dance choreographing, welcome mutations for refinement. In this work, we have edited the timing of the dance steps according to the music. A different area that we are exploring is par de deux movements (dance for two) that can be improvised on stage.

3. The computer dance animation can be integrated into the performance to create interesting effects on stage. Incorporating visual images with dance is not new. Cunningham has worked with visual artists such as Robert Rauschenbuerg, Andy Warhol and Jasper Johns in his dances [1]. This is an area that we will pursue in the future.

References

1. Celant, G. (ed.): Merce Cunningham. Fundacio Antoni Tapies, Barcelona (1999).
2. Cunningham, G.: The Dancer and the Dance/conversations with Jacqueline Lesschaeve New York and London: Marion Boyars, (1985); second (paperback) edition (1991).
3. Johnson, P. E. and Lancaster, A: Swarm User Guide (2002). http://www.santafe.edu/projects/swarm/swarmdocs/userbook/userbook.html
4. Life Forms User Guide. Credo International Inc. (1999).
5. Kostelanetz, R. (ed): Merce Cunningham/Dancing in Space and Time; essays 1944-1992 Pennington NJ: a cappella books (1992)
6. McDonagh, D.: Martha Graham: a Biography. Praeger Publishers, New York (1973).
7. Minar, N., Burkhart, R. T.; Langton, C. and Askenazi, M.:The Swarm Simulation Sys-tem: A Toolkit for Building Multi-Agent Simulations. .SFI Working Paper 96-06-042. (1996). http://www.swarm.org/archive/overview.ps.

The Interpretation of Dreams
An Explanation of the Electric Sheep Distributed Screen-Saver

by Scott Draves

The name *Electric Sheep* comes from Philip K. Dick's novel *Do Androids Dream of Electric Sheep*. It realizes the collective dream of sleeping computers from all over the Internet.

Electric Sheep is a distributed screen-saver that harnesses idle computers into a render farm with the purpose of animating and evolving artificial life-forms.

The project is an attention vortex. It illustrates the process by which the longer and closer one studies something, the more detail and structure appears.

I. The Client

When the software is activated, the screen goes black and an animated 'sheep' appears. In parallel, the screen-saver client contacts the server and joins the distributed computation of new sheep, an idea inspired by the SETI@home project [1].

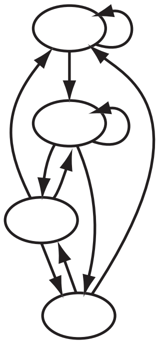
The screen-saver is a window into a visual space shared among all users. Clients render JPEG frames and upload them to the server. When all the frames are ready the server compresses them into an MPEG animation. Each animation is the phenotype of an artificial organism, an "electric sheep". Clients download the MPEG sheep and display them one after another in a continuous, ever-changing sequence.

About once every fifteen minutes a new sheep is born and distributed to all active clients. Each sheep is an animated fractal flame [2]. Its shape is specified by a string of 84 real numbers - a genetic code of sorts. Some of the codes are chosen at random by the server with heuristics to avoid malformed sheep, somewhat like spontaneous abortion. The rest are derived from the current population according to a genetic algorithm with mutation and cross-over.

II. The Server

The server has a web interface for people in addition to the one used by clients. It allows users to see and download the currently living sheep as well as monitor the rendering of new ones. Clients can identify themselves with a nickname and URL and see exactly which frames are theirs. The server generates rankings of nicknames and IP addresses by the number of frames contributed. Users can visit each other's web pages and find out who else is in the community.

Normally electric sheep is very reliable and runs for weeks without assistance, but with new versions come new bugs, and at these times the ability to tweak the server live and online is essential to keeping the flock healthy. By entering a password, a user can become an administrator and delete bad frames, entire sheep, or block clients by address. An administrator can also inject a particular genetic code into the system, for example, to resurrect a sheep from the code stored in a previously captured MPEG file.



III. Life, Death, and Interpolation

A sheep's life is finite. I only have enough disk quota to keep about thirty alive on the server. Old sheep are deleted without a trace. Users may vote for a sheep by pressing the up arrow key when that sheep is displayed on their screen [3]. Popular sheep live longer, and are more likely to reproduce. Hence, the users' preferences provide the fitness function for an aesthetic evolutionary algorithm.

The parameter space of sheep is continuous, and the server generates smooth transitions between sheep by interpolating in the genetic space.

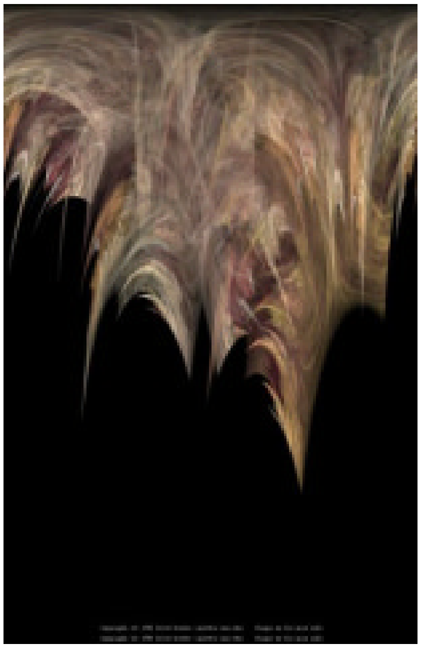
The set of animations on the client form a graph, as illustrated by the diagram above. Each arrow represents an animation. The nodes represent key-frames. A sheep animation is an arrow with the same key-frame at its head and its tail, because sheep are loops. The client plays the animations by following the arrows head to tail and branching and to seek out new territory.

IV. Measurements and Statistics

Clients typically store about 100 sheep totalling 7 minutes of animation and taking 250 megabytes of disk space. The server uses a free MPEG2 video encoder at a resolution of 640 by 480 pixels and 5 megabits per second.

The high resolution sheep available from the web pages and in the video documentary were born on the sheep server, then the parameters were tweeked to increase quality, and finally they were re-rendered and compressed off-line to avoid MPEG compression artifacts.

In ten days at the end of October 2001, clients from 650 unique IP addresses contributed frames to the server. Multiple users may share an address, and no attempt is made to uniquely identify clients, so the real user count is unknown. At that time about 150 clients were participating in the render farm at any one time. In the first 12 days of March 2003, clients from 4900 unique IP addresses downloaded animations from the two operational sheep servers (the second server supports legacy clients).



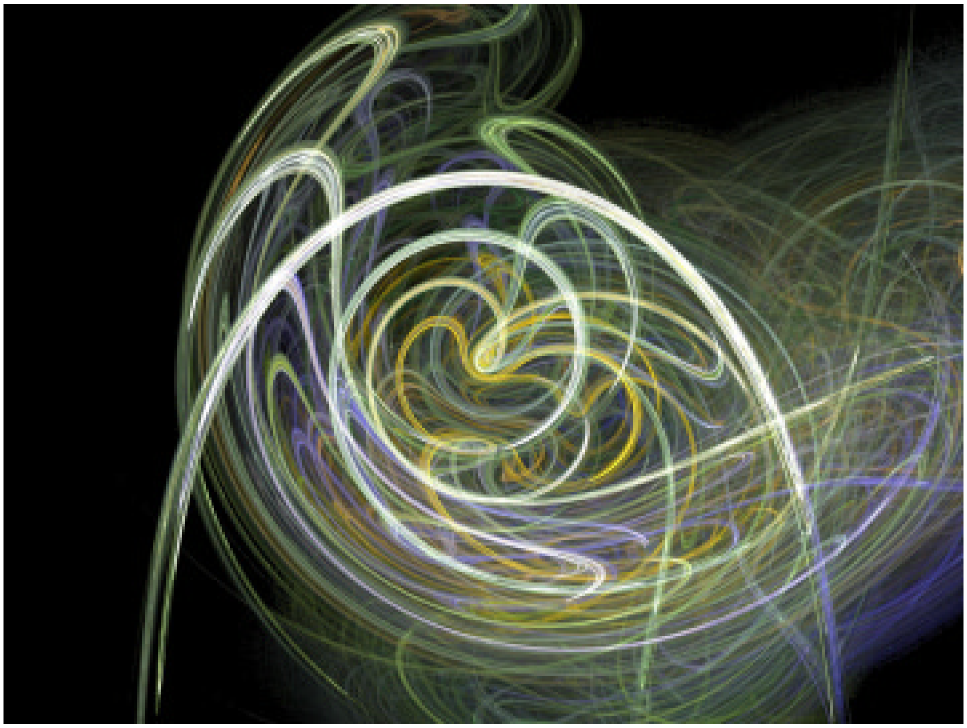
V. Development

From August 1999 when it was created until October 2001, the Electric Sheep client only ran on Linux. At that time Matt Reda released a Mac OS X client, and the number of clients quickly doubled. Despite many requests, several promises, and one near miss, no working Microsoft Windows version has appeared. The user base is also limited to those with high-bandwidth, always-on connections to the Internet such as DSL, cable-modem, or university or corporate networks.

In October 2002 the domain name was hijacked by a competing "electric sheep" site. Fortunately, after a hacking and legal scuffle, the domain has been returned and the site is back in operation, though the user base suffered a set-back.

In February 2003 version 2.3 finally left beta, making near-DVD quality full-screen graphics a reality. It also adopted a compressed XML protocol to save bandwidth and allow more freedom to change without breaking backwards compatibility.

Both clients and the server are open source and there is a developer community as well as a user community. The whole system, centered around the electricsheep.org web site, has its own buzz. The users and developers exchange messages by the discussion forum and email, and clients and servers exchange images and animations. There is an evolving ecology of agents, codes, and protocols.

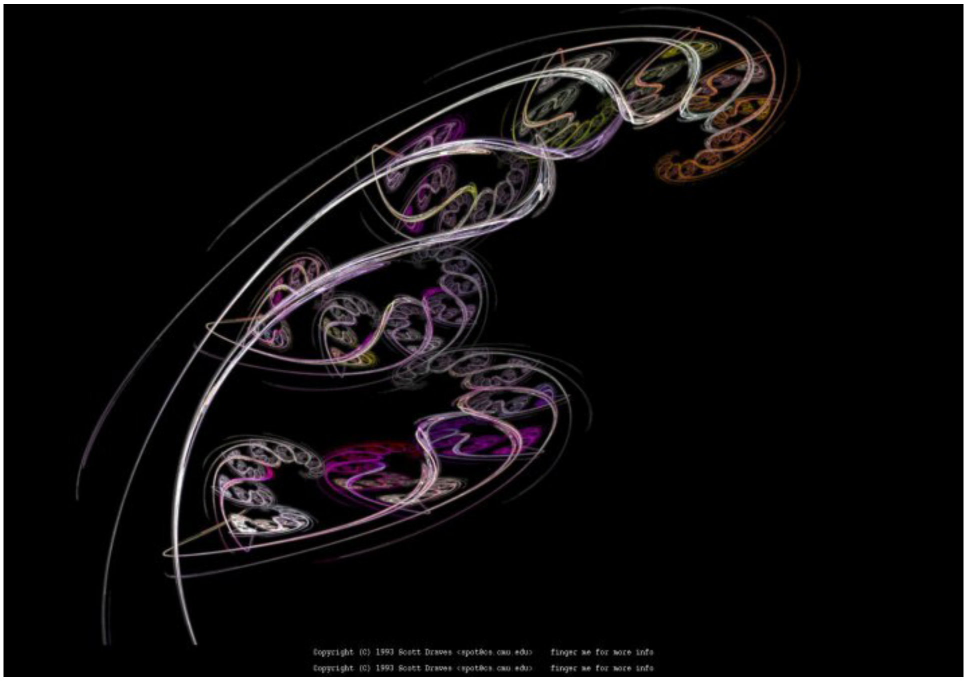


VI. The Vortex

Electric sheep investigates the role of experiencers in creating the experience. If nobody ran the client, there would be nothing to see. Eons ago, tiny irregularities in our universe became centers of accretion and eventually grew into stars. A parallel process unfolds in cyberspace. It starts with an idea.

The sheep system exhibits increasing returns on each of its levels. As more clients join, more computational muscle becomes available, and the resolution of the graphics may be increased, either by making the sheep longer, larger, or sharper. The more people who participate, the better the graphics look.

Likewise, as developers focus more of their attention on the source code, the client and server themselves become more efficient, grow new features, and are ported into new habi


Copyright (C) 1993 Scott Draves <spot@cs.cmu.edu>   finger me for more info
Copyright (C) 1993 Scott Draves <spot@cs.cmu.edu>   finger me for more info

tats. The project gains momentum, and attracts more developers.

And as more users vote for their favorite sheep, the evolutionary algorithm more quickly distills randomness into eye candy.

Perhaps attention acts on information the same way gravity acts on mass: attraction begets attraction and a positive feedback loop is formed.

VII. The Future

Electric sheep is open-ended and very much a work in progress. For example, the server is currently a bottleneck because it delivers large MPEGs to so many clients. But if clients act as servers and become a true peer-to-peer network, the bandwidth load could be distributed much as the computational load already is.

The architecture is not specific to fractal flames, and the protocol should support multiple alternate renderers. I am seeking collaborators to contribute their own generative animation software.

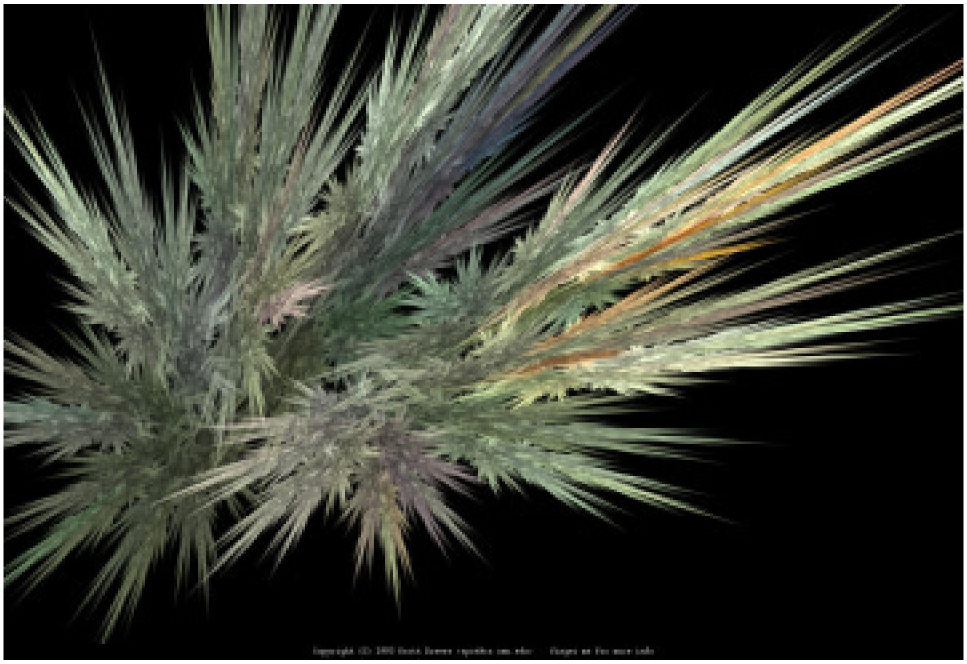
I believe the free flow of code is an increasingly important social and artistic force. The proliferation of powerful computers with high-bandwidth network connections forms the substrate of an expanding universe. The electric sheep and we their shepherds are colonizing this new frontier.

Electricsheep.org

Notes

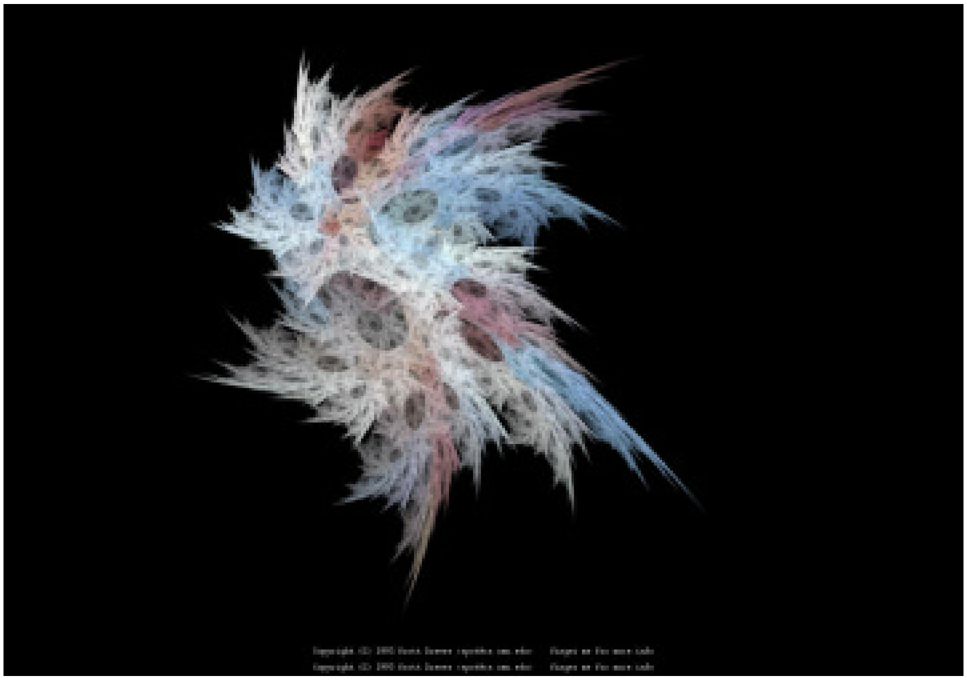
[1] SETI@home searches for a signal from extra-terrestrials in radio-telescope data. It consists of a screen-saver client that is downloaded and installed by users all over the world, and a server that divides-up the data among the clients and collects the results. It puts idle computers to work. SETI@home is the original distributed screen-saver, and its architecture is the inspiration for Electric Sheep's.

[2] *Fractal flames* are the output of a particular Iterated Function System (IFS) fractal rendering algorithm created by the author in 1992. Each image is a histogram of a two-dimen



sional strange attractor. The flame algorithm contains three innovations: (a) It uses a collection of special functions that are composed with the usual affine matrices, (b) the intensity of each pixel is proportional to the logarithm of the density of the attractor rather than a linear relationship, and (c) the color is determined by appending a third coordinate to the chaotic system and looking it up in a palette. Great care is taken to correctly anti-alias the image, both spatially and temporally (with motion blur). Flame is designed to produce images without artifacts, and to reveal as much of the information contained in the attractor as is possible. For more information, see FLAM3.COM.

3. Pressing the up or down arrow key transmits a vote for or against the currently displayed sheep. The server's web interface also has voting controls. In Linux, voting by key-press requires a special version of xscreensaver (part of the gnome desktop interface) to work, so it is not widely (if at all) deployed.



Membership Form

YLEM Yearly Membership Includes:
Membership Directory: An annual publication which you are listed with approximately 250 other artists of new art forms.
Journals: The bi-monthly YLEM Journal contains articles on numerous topics along with news of members.
Forums: YLEM presents bi-monthly forums at San Fracsico's Exploratorium, curates shows, and arranges special site visits.
Web Site Listing: The YLEM web site includes a link to member web sites.

Name

Business Name

Adress

Home Phone

Fax

Work Phone

E-Mail

Web Site

__New or __Continuing member

Please describe your work and/or interests in 30 words or less as you would like it to appear in the directory (art, art-science or technology-related interests, services, etc.). Use extra paper if necessary.

Privacy:
__ Please do not include me in the web site directory.
__ Please do not include me in the printed directory.
__ Please do not include my name when the ylem mailing list is sold to other members.

One-Year Membership Rates

US Individual	$40
US Institutional	$60
US Student or Senior	$25
Contributing Member	$100
Donor Member	$300
Patron Member	$500
Cyber Star Member	$1000

To join online, go to the YLEM website
www.ylem.org

Canada/Mexico add $5 (USD) all other countries add $25 (USD) to US rates.
(US currency only). Please mail in a check or money order payable to Ylem, P.O. Box 749 Orinda CA 94563.
Membership includes next edition of the Directory.
For more information contact:
Eleanor Kent (membership)
ekent@well.com
Tel. 415 647-8503

YLEM

n. pronounced eylum, 1. a Greek word for the exploding mass from which the universe emerged.

An international organization of artists, scientists, authors, curators, educators, and art enthusiasts who explore the Intersection of the arts and sciences. Science and technology are driving forces in the contemporary culture. YLEM members strive to bring the humanizing and unifying forces of art to this arena. YLEM members work in new art media such as Computers, Kinetic Sculpture, Interactive Multimedia, Holograms, Robotics, 3-D Media, Film, and Video.

YLEM
P.O. Box 749
Orinda, CA
94563
USA

Contact Information

YLEM Main Contact
P.O. Box 2590
Alameda, CA 94501
http://www.ylem.org
ekent@well.com

YLEM Membership Issues
members@ylem.org

YLEM Journal Items
Loren Means
149 Evelyn Way
San Francisco, CA 94127 USA
lorenmea@pacbell.net